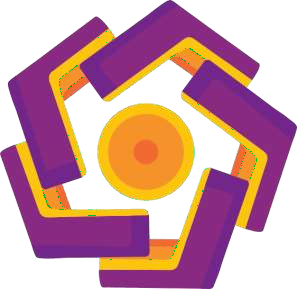
LAPORAN PRAKTIKUM

SISTEM OPERASI

“Bash Shell”

Dosen Pengampu :   
Ferian Fauzi Abdulloh M.Kom



Disusun oleh:

Nama : Muhammad Rifki

NIM : 22.11.4636

Kelas : 22 BCI

UNIVERSITAS AMIKOM YOGYAKARTA 2022/2023

# Percoban 1

# 

# Analisa:

# Perintah ls -a digunakan untuk menampilkan seluruh isi directory home termasuk file yang hidden/tersembunyi. Pada perintah $ more .bash\_profile, tidak terdapat file dan direktorinya. Hal ini terjadi untuk beberapa versi ubuntu. Namun pada umumnya file ini berisi konfigurasi bash shell.

# Percoban 2

# 

# Percobaan 2: Menggunakan Feature History Bash

# 1.Bash shell menyimpan “history” perintah yang digunakan sebelumnya. Anda dapat mengaksis history dalam beberapa cara. Cara paling mudah adalah menggunakan Panah Atas. Maka perintah sebelumnya akan ditampilkan.

# Hasil:

# 

# 

# Analisa:

# Ketika menekan panah atas, secara otomatis menampilkan perintah yang sebelumnya pernah ditulis.

# 

# 2.      Berikutnya, berikan Bash shell beberapa perintah untuk diingat. Masukkan perintah berikut dan tekan Enter pada setiap baris.

# $ cd

# $ ls –l /etc

# $ ls –l

# $ whoami

# $ who

# Hasil:

# 

# 

# 

# 

# 

# 

# Analisa:

# Perintah di atas digunakan untuk diingat dan akan dipanggil pada nomor 3 untuk mengecek history. Perintah cd digunakan untuk masuk suatu directory. Perintah ls -l /etc digunakan untuk me-list isi direktori etc. Perintah ls -l digunakan untuk me -list atau melihat daftar direktori home. Perintah who dan whoami digunakan untuk menampikan user dan user yang sedang aktif.

# 

# 3.   Untuk memeriksa apakah perintah ini ditambahkan pada history, dapat menggunakan perintah history untuk melihat semua perintah yang pernah dimasukkan.

# $ history

# Hasil:

# 

# 

# Analisa:

# Perintah history digunakan untuk menampilkan history input yang pernah dimasukkan ke terminal berupa input keyboard. Perintah pada nomor 2 yang telah diinputkan juga masuk ke dalam history.

# 

# 4.      Anda dapat memilih perintah sebelumnya dengan menggunakan Panah Atas, tetapi hal ini tidak efisien untuk perintah yang semakin bertambah banyak. Cara yang mudah menggunkaan nomor pada perintah history atau mencarinya. Untuk memilih dan mengeksekusi perintah dengan nomor, masukkan kunci! diikuti nomor perintah.

# $ ! <Nomor Perintah>                   Contoh: !780

# Hasil:

# 

# 

# 

# 

# Analisa:

# Perintah di atas digunakan untuk memasukkan input yang sama sesuai dengan nomor yang ditunjuk. Nomor tersebut dapat dilihat ketika perintah history dijalankan. Ketika memasukkan perintah nomor 647, yang dimana merupakan perintah who. Maka selanjutnya perintah itupun berjalan.

# 

# 5.      Anda dapat mencari perintah dengan menyertakan perintah yang diinginkan. Misalnya !?etc?! akan menjalankan perintah ls –l /etc yang sebelumnya digunakan.

# $ !?etc?

# Hasil:

# 

# 

# Analisa:

# Perintah !?etc? adalah perintah yang digunakan untuk mencari perintah yang ada hubungannya dengan etc dan yang pernah digunakan. Sedangkan kita pernah menggunakan perintah ls -l /etc maka perintah itupun berjalan karena kita memanggilnya dengan kata kunci etc.

# 

# 6.      Kemudian gunakan perintah history, maka akan terlihat perintah ls –l /etc yang kedua dan bukan !?etc?

# $ history

# Hasil:

# 

# 

# 

# 

# Analisa:

# Pada gambar di atas, dapat dilihat bahwa walaupun kita menginputkan perintah !?etc? tetapi yang berada dalam history adalah ls -l /etc karena yang berjalan adalah program tersebut. Dengan kata lain !?etc? bukanlah perintah lain tetapi merupakan pancingan untuk memanggil ls -l /etc.

# 

# 7.      Apabila string tidak ditemukan pada perintah history maka akan terdapat pesan error.

# $ !?wombat99?

# Hasil:

# 

# 

# Analisa:

# Ketika perintah !?wombat99? dijalankan, terdapat pesan error. Hal ini terjadi karena kita memasukkan kata yang belum pernah dimasukkan ke terminal sebelumnya.

# 

# 8.      Jika diketikkan !who maka yang dijalankan adalah perintah who. Tetapi bila Anda ketikkan !whoa maka yang dijalankan adalah perintah whoami.

# $ !who

# $ !whoa

# Hasil:

# 

# 

# Analisa:

# Perintah !who sama saja dengan perintah who, tanda ! hanya untuk jika terjadi salah mengetikan perintah.  Misal, kita ingin memasukan perintah whoami, tetapi ternyata kita salah ketik menjadi whoa. Tapi karena kita sudah menggunakan tanda ! maka yang keluar tetaplah output dari perintah whoami.

# 

# 9.      Anda bisa menggantikant string pada perintah history, terutama pada perintah yang panjang. Misalnya ketik cat /bin/bash | strings | grep shell | less dan tekan Enter. Maka akan menampilkan semua string pada file /bin/bash yang berisi kata “shell”. Untuk keluar tekan q. Jika ingin menampilkan kata “alias”, maka Anda tidak perlu mengetik perintah yang panjang lagi, tetapi cukup ketik ^shell^alias^ dan tekan Enter maka akan menggantikan kata “shell” dengan “alias”.

# $ cat /bin/bash | strings | grep shell | less

# $ ^shell^alias^

# Hasil:

# 

# 

# 

# 

# 

# 

# 

# 

# Analisa:

# Perintah cat /bin/bash | strings | grep shell | less dimaksudkan untuk mencari kata shell dengan cara membaca /bin/bash. Sedangkan perintah ^shell^alias^ digunakan untuk mengganti kata shell pada perintah sebelumnya dan diganti dengan kata alias. Perintah tersebut sama saja dengan perintah cat /bin/bash | strings | grep shell | less, tetapi untuk menyingkat hal tersebut cukup menggunakan perintah ^shell^alias^.

# 

# Percobaan 3: Mengubah Feature History Bash

# 1.      Bash shell akan menyimpan perintah history meskipun telah log out dan log in kembali. File .bash\_history menyimpan file history yang terdapat pada home directory.

# $ cd

# Hasil:

# 

# 

# Analisa:

# Perintah cd digunakan untuk masuk ke dalam direktori.

# 

# 2.      Lihat beberapa baris pada file .bash\_history dengan ketik tail .bash\_history dan tekan Enter. File ini bukan file yang up to date.

# $ tail .bash\_history

# Hasil:

# 

# 

# Analisa:

# Pada perintah di atas, yang ditampilkan adalah perintah terakhir yang diinputkan sebelum komputer kita restart atau matikan. Perintah history sampai cat /bin/bash | strings | grep shell | less adalah perintah yang diinputkan sebelum saya mematikan komputer.

# 3.      Ketik history dan tekan Enter. Maka akan terlihat baris terakhir adalah perintah history dan baris sebelumnya adalah tail .bash\_history. Perintah history bersifat up to date, karena disimpan pada memory sistem.

# $ history

# Hasil:

# 

# 

# 

# 

# Analisa:

# Perbedaaan perintah history dengan tail .bash\_history adalah jika perintah history menampilkan perintah yang kita inputkan terakhir/paling baru. Sedangkan perintah tail .bash\_history menampilkan perintah yang terakhir kita masukkan sebelum komputer kita matikan/restart (bukan yang terbaru/up to date).

# 

# 4.      Ketik perintah berikut

# $ echo ‘Ini perintah saya’

# Hasil:

# 

# 

# Analisa:

# Perintah echo digunakan untuk mencetak teks “Ini perintah saya”.

# 

# 5.      Log out dan log in kembali sebagai user yang sama. Ketik history dan tekan Enter. Maka perintah echo ’Ini perintah saya’ akan berada pada baris terakhir. Lihat file .bash\_history, maka perintah tsb akan terdapat pada file .bash\_history.

# $ history

# $ tail .bash\_history

# Hasil:

# 

# 

# 

# 

# 

# 

# Analisa:

# Perintah echo “Ini perintah saya” masuk ke dalam history yang terakhir. Namun perintah echo “Ini perintah saya” tidak masuk ketika perintah tail .bash\_history dijalankan. Hal ini terjadi karena saya belum sempat untuk mematikan komputer saya sehingga perintah echo pada tail .bash\_history tidak masuk.

# 

# 6.      Ketik history|less untuk melihat perintah history terakhir pada screen. Tekan spacebar untuk melihat file lebih banyak. Untuk keluar tekan q

# $ history|less

# Hasil:

# 

# 

# 

# 

# Analisa:

# Perintah di atas sama saja dengan melihat perintah history terakhir pada screen.

# 

# 7.      Untuk melihat berapa banyak perintah history yang ada pada file ketik berikut dan output yang keluar serupa di bawah ini

# $ wc –l .bash\_history

# 1000 .bash\_history

# Hasil:

# 

# 

# Analisa:

# Perintah wc -l digunakan untuk menampilkan jumlah history pada file.

# 

# 8.      Output menunjukkan bahwa 1000 perintah history disimpan pada file history. Untuk melihat jangkauan (limit) perintah history digunakan variabel HISTSIZE. Untuk melihat jangkauan history ketik sebagai berikut

# $ set|grep HISTSIZE

# Hasil:

# 

# 

# Analisa:

# Perintah di atas digunakan untuk melihat jangkauan (limit) atau kuota perintah history yang dapat disimpan bash.

# 

# 9.      Bila ingin memperbesar jangkauan file history, maka ubahlah variabel HISTSIZE pada skrip startup yang disebut .bashrc pada home directory.

# $ echo ‘HISTSIZE=5000’ >> .bashrc

# Hasil:

# 

# 

# Analisa:

# Maksud dari perintah di atas adalah untuk memperbesar limit penyimpanan perintah pada history dari 1000 menjadi 5000.

# 

# 10.  Log out dan log in kembali sebagai user yang sama. Lihat perubahan variabel HISTSIZE.

# $ set|grep HISTSIZE

# Hasil:

# 

# 

# Analisa:

# Pada gambar di atas dapat dilihat bahwa kita telah berhasil mengubah histsize dari 1000 menjadi 5000.

# 

# 11.  Ketikkan perintah history beberapa kali, maka perintah ini akan disimpan pada BASH history meskipun yang diketikkan perintahnya sama.

# Hasil:

# 

# 

# Analisa:

# Perintah di atas, jika perintah history di ketik beberapa kali, maka perintah ini hanya disimpan pada bash sehingga hasilnya tetap sama walaupun perintah diketik beberapa kali.

# 

# 12.  Anda dapat melakukan konfigurasi BASH agar tidak menambah perintah ke history jika perintah yang diketikkan sama dengan sebelumnya. Hal ini dilakukan dengan menambahkan variabel HISTCONTROL dan diberikan nilai ignoredups pada file .bashrc

# $ echo ‘HISTCONTROL=ignoredups’ >> .bashrc

# Hasil:

# 

# 

# 

# 

# Analisa:

# Saya telah mengetikkan history banyak sekali, dan setelah itu kita menginputkan perintah tersebut dengan maksud agar perintah history yang saya masukkanhanya disimpan dalam bash.

# 

# 13.  Log out dan log in kembali sebagai user yang sama. Ketikkan history beberapa kali dan perhatikan berapa kali history muncul.

# Hasil:

# 

# 

# Analisa:

# Dapat dilihat bahwa perintah history yang saya masukkan hanya tersimpan satu yaitu pada nomor 676 padahal saya menginputkan perintah history berkali kali.

# 

# Percobaan 4: Mengubah Prompt Shell

# 1.      Prompt Bash shell dikonfigurasi dengan men-setting nilai variabel PS1. Selain menampilkan string statik sebagai prompt, Anda dapat menampilkan menjadi dinamis. Contohnya, apabila ingin menunjukkan *current directory* atau *current time*. Ketik PS1=’\t:’ dan tekan Enter untuk menampilkan waktu sistem dalam format 24 jam sebagai prompt Bash. Format dalam HH:MM:SS

# $ PS1=’\t:’

# Hasil:

# 

# 

# Analisa:

# Perintah di atas digunakan untuk mengganti prompt dari egypt@egypt-virtualBox: menjadi waktu.

# 

# 2.      Untuk menampilkan format 12 jam dengan indikator am dan pm ketik sebagai berikut:

# $ PS1=’\t:’

# Hasil:

# 

# 

# Analisa:

# Ternyata outputnya sama seperti sebelumnya, yaitu tetap dalam format 24 jam.

# 

# 3.      Kebanyakan orang menginginkan prompt Bash menampilkan *current working directory*. Direktory dapat ditampilkan dalam bentuk keseluruhan path atau hanya nama direktory. Karakter \w menampilkan hanya nama direktory. Jika *current directory* adalah home directory, maka tampil prompt

# ~:

# $ PS1=’\w:’

# Hasil:

# 

# 

# Analisa:

# Perintah tersebut digunakan untuk menampilkan prompt bash seperti pada gambar.

# 

# 4.      Ketik cd /usr/sbin untuk melihat prompt /usr/sbin:

# $ cd /usr/sbin

# Hasil:

# 

# 

# Analisa:

# Perintah cd /usr/sbin digunakan untuk melihat prompt /usr/bin:

# 

# 5.      Ketik PS1=’\W:’ untuk melihat prompt sbin:

# $ PS1=’\W:’

# Hasil:

# 

# 

# Analisa:

# Perintah di atas digunakan untuk melihat prompt sbin.

# 

# 6.      Ada beberapa prompt BASH lain yang dapat diubah, yaitu PS2, PS3 dan PS4. Prompt PS2 digunakan sebagai prompt sekunder. Untuk melihat bagaimana penggunaannya, ketik echo ’Hello (tanpa diakhiri penutup quote) dan tekan Enter. Simbol lebih besar dari (>) akan muncul. Hal ini memberitahukan bahwa BASH menunggu Anda menyelesaikan perintah. Ketik penutup quote (‘) dan tekan Enter. Perintah ini akan menyelesaikan prompt PS2, kata ”Hello, ” muncul diikuti dengan prompt PS1 pada baris baru.

# $ echo ’Hello

# >’

# Hasil:

# 

# 

# Analisa:

# Simbol lebih besar dari (>) muncul. Hal ini memberitahukan bahwa BASH menunggu Anda menyelesaikan perintah. Lambang > menunjukan adanya instruksi yang belum selesai dari instruksi yg saya masukan yaitu echo, karena saya memulai dengan ‘ tapi blm saya tutup ,maka ketika diketik enter hasilnya adalah seperti digambar.

# 

# 7.      Anda dapat mengubah prompt PS2 seperti mengubah prompt PS1. Ketik perintah berikut: $ PS2=’Selesai memasukkan perintah Anda:’

# Hasil:

# 

# 

# Analisa:

# Ketik penutup quote (‘) dan tekan Enter. Perintah ini akan menyelesaikan prompt PS2, kata ”Hello”, muncul diikuti dengan prompt PS1 pada baris baru.

# 

# 8.      Kemudian ketik echo ’Hello (tanpa diakhiri penutup quote) dan tekan Enter. Pada baris berikutnya akan muncul Selesai memasukkan perintah Anda:. Kemudian ketikkan penutup quote (’) dan tekan Enter. Jika perintah selesai, maka kata Hello akan muncul diikuti prompt PS1 pada baris baru.

# $ echo ’Hello

# Selesai memasukkan perintah Anda:’

# Hasil:

# 

# 

# Analisa:

# Perintah ini digunakan untuk menampilkan prompt PS1.

# 

# 9.      Prompt BASH dapat ditampilkan berwarna dengan melakukan setting *colorsetting string*. Sebagai contoh, prompt BASH di-set dengan \w\$, akan menampilkan *current working directory* yang diikuti $ (atau # jika anda login sebagai root). Untuk setting warna menjadi biru ketikkan berikut:

# $ PS1=’\033[0;34m\w\$ \033[0;37m’

# Hasil:

# 

# 

# Analisa:

# Perintah di atas digunakan untuk mengganti warna prompt bash menjadi warna biru. Keterangan untuk mengubah warna sebagai berikut:

# 30=hitam, 31=merah, 32=hijau, 34=biru, 35=ungu, 36=cyan, 37=putih.

# 

# 10.  Untuk mendapatkan prompt warna merah ketikkan berikut:

# $ PS1=’\033[0;31m\w\$ \033[0;37m’

# 30=hitam, 31=merah, 32=hijau, 34=biru, 35=ungu, 36=cyan, 37=putih.

# Hasil:

# 

# 

# Analisa:

# Perintah diatas dimaksudkan untuk mengganti warna prompt bash menjadi warna merah. Keterangan untuk mengubah warna sebagai berikut:

# 30=hitam, 31=merah, 32=hijau, 34=biru, 35=ungu, 36=cyan, 37=putih.

# 

# 11.  Bila menginginkan beberapa warna, ketikkan perintah berikut:

# $ PS1=’\033[0;31m\w\033[0;32m\$ \033[0;37m’

# Hasil:

# 

# 

# Analisa:

# Perintah di atas digunakan untuk mengganti warna prompt bash menjadi beberapa warna. Keterangan untuk mengubah warna sebagai berikut:

# 30=hitam, 31=merah, 32=hijau, 34=biru, 35=ungu, 36=cyan, 37=putih.

# 

# 12.  Anda bisa menampilkan atribut visual seperti lebih terang, berkedip dan warna kebalikannya. Untuk menampilkan prompt yang lebih terang, atribut control diganti 1, seperti perintah berikut:

# $ PS1=’\033[1;34m\w\033[1;32m\$ \033[0;37m’

# Hasil:

# 

# 

# Analisa:

# Perintah di atas digunakan untuk mengubah warna pada prompt bash. Atribut kontrol 1 dimaksudkan untuk membuat warna menjadi lebih terang.

# 

# 13.  Untuk menampilkan prompt dengan warna berkebalikan, atribut control diganti 7, seperti perintah berikut:

# $ PS1=’\033[7;34m\w\033[7;32m\$ \033[0;37m’

# Hasil:

# 

# 

# Analisa:

# Atribut control angka 7 digunakan untuk membuat warna kebalikannya.

# 

# 14.  Untuk menampilkan prompt berkedip, atribut control diganti 5, seperti perintah berikut:

# $ PS1=’\033[5;34m\w\033[5;32m\$ \033[0;37m’

# Hasil:

# 

# 

# 

# 

# Analisa:

# Atribut control angka 5 digunakan agar prompt bashnya berkedip.

# 

# Percobaan 5: Menambahkan otomatisasi ke Prompt Shell

# 1.      Pastikan Anda berada di home directory

# $ cd ~

# Hasil:

# 

# 

# Analisa:

# Perintah cd digunakan untuk masuk ke dalam direktori home.

# 

# 2.      Buatlah skrip sederhana untuk mengurut daftar file. Anda dapat menggunakan teks editor, tetapi karena hanya satu baris, gunakan perintah echo untuk membuat file.

# $ echo ’sort ~/list > ~/r13; mv ~/r13 ~/list’ > ~/sorter

# Hasil:

# 

# 

# Analisa:

# Perintah diatas merupakan perintah menambahkan otomatosasi ke Prompt Shell. Pastikan sedang berada di home direktori, kemudian membuat skrip sederhana untuk mengurut daftar file. Karena hanya satu baris, gunakan perintah echo untuk membuat file. Perintah echo ‘sort ~/list > ~/r13; mv ~/r13 ~/list’ > ~/sorter untuk membuat deteran file terurut.

# 

# 3.      Buatlah file skrip diatas menjadi file executable

# $ chmod +x sorter

# Hasil:

# 

# 

# Analisa:

# Perintah tersebut digunakan untuk membua file skrip menjadi file executable.

# 4.      Jalankan program sorter diatas setiap shell Bash menampilkan prompt PS1. Untuk melakukannya, buatlah variable PROMPT\_COMMAND dimana nilainya adalah nama dari program sorter.

# $ PROMPT\_COMMAND=~/sorter

# Hasil:

# 

# 

# Analisa:

# Untuk menjalankan program sorter diatas setiap shell Bash menampilkan prompt PS1.

# 

# 5.      Ketikkan echo ’John Smith:13001’>>list dan tekan Enter. Jika file list tidak ada, akan dibuat secara otomatis, tetapi jika sudah ada, string ’John Smith:13001’ akan ditambahkan. $ echo ’John Smith:13001’>>list

# Hasil:

# 

# 

# Analisa:

# Percobaan ini dimaksudkan untuk memasukkan John Smith:13001 kedalam file list.

# 

# 6.      Ketik cat list dan tekan Enter. Maka Anda akan melihat isi file list. Pada saat ini, file mungkin mempunyai hanya satu baris sehingga tidak dapat dilihat apakah file sudah terurut.

# $ cat list

# Hasil:

# 

# 

# Analisa:

# Perintah cat list digunakan untuk membaca file list atau melihat isi file list yang telah dibuat.

# 

# 7.      Masukkan bebe rapa perintah serupa dengan point 5 tetapi dengan nama dan nomor yang berbeda. Kemudian ketik cat list dan tekan Enter.

# $ echo ’Anita:13002’>>list

# $ echo ’Samantha:13003’>>list

# $ echo ’Patrik:13004’>>list

# $ echo ’Sponge Bob:13005’>>list

# $ echo ’Lisa:13006’>>list

# $ echo ’Squid:13007’>>list

# Hasil:

# 

# 

# Analisa:

# Perintah diatas dimaksudkan kita memasukkan nama-nama tersebut kedalam file list. Perintah cat digunakan untuk menampilkan isi file list secara urut sesuai nama abjad.

# 

# 8.      Apabila Anda tidak menginginkan Shell Bash menampilkan file terurut sepanjang waktu, Anda tidak perlu menambahkan variable PROMPT\_COMMAND=~/sorter pada file konfigurasi seperti .bashrc. Bila Anda ingin BASH berhenti menjalankan program sorter, maka ketikkan variable PROMPT\_COMMAND= dan tekan Enter atau log out dan login kembali.

# $ PROMPT\_COMMAND=

# Hasil:

# 

# 

# Analisa:

# Perintah di atas digunakan untuk menghentikan program sorter.

# 

# Percobaan 6: Membuat Bash-script dan menjalankannya

# 1.      Membuat file p1.sh

# $ vi p1.sh

# echo “Program bash Script”

# Hasil:

# 

# 

# 

# 

# Analisa:

# Pertama-tama membuat file p1.sh menggunakan editor. Untuk memasukkan suatu kata, terlebih dahulu tekan huruf i. Kemudian kita masukkan di dalamnya echo “Program bash Script”. Untuk keluar, bisa menekan esc, lalu tekan :, dan dilanjutkan dengan mengetik wq! dan Enter.

# 

# 2.      Mengubah program menjadi executable

# $ ls –l p1.sh

# $ chmod +x p1.sh

# $ ls –l p1.sh

# Hasil:

# 

# 

# Analisa:

# Perintah $ ls –l p1.sh digunakan untuk menampilkan file p1.sh. Sedangkan perintah $ chmod +x p1.sh dimaksudkan agar file p1.sh dapat dijalankan di linux.

# 

# 3.      Menjalankan script

# $ bash p1.sh

# $ sh p1.sh

# $ . p1.sh

# $ ./p1.sh

# Hasil:

# 

# 

# Analisa:

# Untuk menjalankan file .sh dapat menggunakan perintah bash, sh, .. dan ./

# 

# 4.      Konvensi dalam pembuatan script shell dinyatakan sebagai #!/bin/bash. Tambahkan pada file p1.sh konvensi tersebut

# $ vi p1.sh

# #!/bin/bash

# echo “Program bash script”

# Hasil:

# 

# 

# 

# 

# Analisa:

# Menambahkan #!/bin/bash pada file p1.sh dengan cara untuk memasukkan suatu kata, terlebih dahulu tekan huruf i. Kemudian kita masukkan di dalamnya “!/bin/bash”. Untuk keluar, bisa menekan esc, lalu tekan :, dan dilanjutkan dengan mengetik wq! dan Enter. Tanda #!/bin/bash  merupakan petunjuk bahwa script file tersebut adalah script bash. Tanda “#” digunakan untuk menambahkan komentar pada script.

# 

# 5.      Buatlah file p2.sh

# $ vi p2.sh

# #!/bin/bash

# echo “Program 2 bash script”

# Hasil:

# 

# 

# 

# 

# Analisa:

# Untuk membuat file p2.sh sama seperti pada p1.sh. Menambahkan #!/bin/bash yang menandakan bahwa script file tersebut adalah file bash. Tanda “#” digunakan untuk menambahkan komentar pada script.

# 

# 6.      Menjalankan beberapa program shell dalam satu baris instruksi yang dipisahkan dengan tanda;

# $ cat p1.sh ; cat p2.sh

# $ ./p1.sh ; ./p2.sh

# Hasil:

# 

# 

# Analisa:

# Kita dapat menjalankan 2 program secara sekaligus dengan cara memberi tanda ;.

# 

# 7.      Menjalankan script sebagai prosees background, sehingga prompt tidak

# 

# Percobaan 7: Job Control

# 1.      Proses foreground

# $ ps x

# Hasil:

# 

# 

# Analisa:

# Perintah ps x berfungsi untuk menampilkan proses pada foreground.

# 

# 2.      Proses background

# $ ps x > hasil &

# Hasil:

# 

# 

# Analisa:

# Untuk menampilkan jumlah proses yang berjalan menggunakan perintah di atas.

# 

# 3.      Setiap job mempunyai PID yang tunggal (unique). Untuk melihat jobs yang aktif

# $ jobs

# Hasil:

# 

# 

# Analisa:

# Perintah jobs digunakan untuk melihat jobs yang aktif.

# 

# 4.      Buatlah file ploop.sh. File ini tidak akan pernah berhenti kecuali ditekan Ctrl-C

# $ vi ploop.sh

# #!/bin/bash

# while [ true ]

# do

# sleep 10

# echo “Hallo”

# done

# Hasil:

# 

# 

# 

# 

# Analisa:

# Pertama-tama, membuat file ploop.sh. Kemudian menuliskan program di atas caranya dengan menekan i, lalu tulis programnya. Untuk keluar, menekan esc, lalu tekan :, dan disertai dengan menulis wq! dan enter untuk menyimpannya. Program tersebut bermaksud untuk setiap 10 detik maka terminal akan mencetak teks “Hallo”. Program dapat dihentikan dengan menekan Ctrl+C.

# 

# 5.      Buatlah file ploop.sh menjadi executable. Jalankan program, akan ditampilkan kata Hallo setiap 10 detik. Untuk keluar program, tekan Ctrl-C (^C)

# $ chmod +x ploop.sh

# $ ./ploop.sh

# Hasil:

# 

# 

# Analisa:

# Perintah chmod +x ploop.sh membuat program menjadi bisa dijalankan dengan linux di terminal. Ketika dijalankan, maka programnya akan menampilkan teks Hallo dan akan terus berjalan sampai dihentikan dengan cara menekan Ctrl+C. Namun, pada program saya, terdapat error karena perintah true tidak ditemukan.

# 

# Percobaan 8: Manipulasi stack untuk Direktori

# 1.      Instruksi dirs digunakan untuk melihat stack direktori, pada output hanya ditampilkan direktori home ~

# $ dirs

# Hasil:

# 

# 

# Analisa: Perintah dirs digunakan untuk melihat stack directory dan menampilkan directory home.

# 

# 2.      Membuat 3 buah direktori

# $ mkdir marketing sales support

# Hasil:

# 

# 

# Analisa:

# Perintah di atas digunakan untuk membuat direktori marketing, sales, dan support.

# 

# 3.      Instruksi dirs digunakan untuk melihat stack direktori, pada output hanya ditampilkan direktori home

# ~ $ dirs

# Hasil:

# 

# 

# Analisa:

# Perintah dirs digunakan untuk melihat stack direktori dan ketika dieksekusi hasilnya menampilkan direktori home.

# 

# 4.      Membuat 3 buah direktori

# 

# Percobaan 9: Alias

# 1.      Alias adalah mekanisme untuk memberi nama alias pada satu atau sekelompok instruksi. Untuk melihat alias yang sudah terdaftar pada system:

# $ alias

# Hasil:

# 

# 

# Analisa:

# Perintah alias digunakan untuk menampilkan perintah alias yang sudah ada pada sistem atau bisa dibilang perintah alias default.

# 

# 2.      Membuat beberapa alias

# $ alias del=’rm –i’

# $ alias h=’history’

# Hasil:

# 

# 

# Analisa:

# Maksud perintah di atas adalah membuat alias yaitu mengganti perintah rm -i dengan alias del, serta mengganti perintah history dengan alias h.

# 

# 3.      Gunakan instruksi hasil alias

# $ ls

# $ del hasil

# $ h | more

# Hasil:

# 

# 

# 

# 

# Analisa:

# Perintah ls digunakan untuk melihat daftar file/direktori, sehingga kita bisa melihat file yang kita hapus sudah terhapus atau belum. Perintah $ del hasil digunakan untuk menghapus direktori hasil. Lalu untuk melihat history dengan tampilan lebih banyak menggunakan h | more. Perintah tersebut bisa dieksekusi karena sebelumnya kita telah mengganti perintah history dengan alias.

# 

# 4.      Untuk menghapus alias gunakan instruksi unalias

# $ unalias del

# $ del files (Terdapat Pesan Kesalahan, mengapa ?)

# Hasil:

# 

# 

# Analisa:

# Perintah unalias del digunakan untuk menghapus alias dari kata del yang sebelumnya adalah rm -i. Sehingga selanjutnya perintah del menjadi kosong dan tidak dialiaskan. Oleh karena itu ketika perintah del files dijalankan, terdapat pesan error karena alias del sebelumnya sudah dihapus.

# 

# LATIHAN:

# 1.      Eksekusi seluruh profile yang ada:

# a.       Edit file profile /etc/profile dan tampilkan pesan sebagai berikut: echo ’Profile dari /etc/profile’

# Hasil:

# 

# 

# 

# 

# 

# 

# Analisa:

# Perintah di atas digunakan untuk mengedit file di etc/profile. Untuk membuat dan mengedit file menggunakan editor, terdapat beberapa perintah yang dapat ditulis seperti nano, vi, dan vim. Namun, disini saya menggunakan perintah nano untuk membuat dan mengedit teks di editor. Setelah editor ditulis, untuk menyimpan filenya menggunakan Ctrl+O dan keluar menggunakan Ctrl+X. Bisa juga menyimpannya dengan cara Ctrl+X, tekan Y, lalu Enter. Kemudian dicek menggunakan su - <user> dan terlihat bahwa profile dari /etc/profile sudah bisa dieksekusi. Yang artinya telah sukses ditambahkan.

# 

# b.      Asumsi nama anda student, maka edit semua profile yang ada yaitu: /home/student/.bash\_profile /home/. student/.bash\_login /home/student/.profile /home/student/.bashrc

# Hasil:

# /home/student/.bash\_profile

# 

# 

# 

# 

# /home/student/.bash\_login

# 

# 

# 

# 

# /home/student/.profile

# 

# 

# 

# 

# /home/student/.bashrc

# 

# 

# 

# 

# Analisa:

# Perintah vi digunakan untuk masuk ke teks editor untuk mengedit file konfigurasi sistem, membuat script shell, membuat program, dan sebagainya.

# 

# c.       Ganti nama /home/student dengan nama anda sendiri. Pada setiap file tersebut, cantumkan instruksi echo, misalnya pada /home/ student/.bash\_profile: echo “Profile dari .bash\_profile”

# /home/egypt/.bash\_profile

# 

# 

# 

# 

# /home/egypt/.bash\_login

# 

# 

# 

# 

# /home/egypt/.profile

# 

# 

# 

# 

# /home/egypt/.bashrc

# 

# 

# 

# 

# Analisa:

# Semua perintah di atas digunakan untuk mengubah semua profile yang sudah ada. Yaitu .bash\_profile, .bash\_login, .profile, dan .bashrc. Untuk mengubah dan mengedit teks dalam editor, caranya sama seperti pada nomor satu menggunakan perintah nano.

# 

# d.      Lakukan hal yang sama untuk file lainnya, sesuaikan tampilan dengan nama file yang bersangkutan.

# Hasil:

# 

# 

# Analisa:

# Perintah editor seperti pada nomo sebelumnya. Fungsi dari setiap file yang diedit akan muncul pada waktu tertentu. Saat mulai session bash akan muncul teks “profile dari .bashrc”. Saat login, akan muncul seperti pada gambar di atas.

# 

# 2.      Jalankan instruksi subtitute user, kemudian keluar dengan perintah exit sebagai berikut:

# $ su student

# $ exit

# kemudian gunakan opsi – sebagai berikut:

# $ su – student

# $ exit

# Jelaskan perbedaan kedua utilitas tersebut.

# Hasil:

# 

# 

# 

# 

# Analisa:

# Perintah di atas adalah perintah yang digunakan untuk pindah ke super user, sistem meminta password admin. Pada perintah tersebut, pada file .bashrc sudah dimodifikasi sebelumnya sehingga instruksi echo terpanggil bila menggunakan perintah su - <user>. Untuk mengakhirinya menggunakan perintah exit.

# Perbedaan dari 2 utilitas tersebut adalah pada su <user> setelah kita memasukkan password maka akan keluar pesan atau perintah dari file .bashrc dan ketika exit maka akan muncul kata exit, sedangkan pada su - <user> akan memunculkan pesan atau perintah dari file .bash\_profile yang berhubungan dengan profile dan ketika exit akan muncul kata logout.

# 

# 3.      Logout

# a.       Edit file .bash\_logout, tampilkan pesan dan tahan selama 5 detik, sebelum eksekusi logout Echo “Terima kasih atas sesi yang diberikan”

# Sleep 5

# Clear

# Hasil:

# 

# 

# 

# 

# 

# 

# Analisa:

# Menambahkan perintah echo untuk menampilkan pesan dan menahannya selama 5 detik dengan perintah sleep. Perintah ini akan dikerjakan saat kita logout sebagai user.

# Ketika kita masuk menggunakan su - <user> maka ketika exit akan dikerjakan echo dan sleep tersebut.

# 

# b.      Edit file .bash\_logout, tampilkan pesan dan tahan selama 4 detik, sebelum eksekusi logout

# Hasil:

# 

# 

# 

# 

# Analisa:

# Menambahkan perintah echo untuk menampilkan pesan dan menahannya selama 4 detik dengan perintah sleep. Perintah ini akan dikerjakan saat kita logout sebagai user.

# Ketika kita masuk menggunakan su - <user> maka ketika exit akan dikerjakan echo dan sleep tersebut.

# 

# 4.      History

# a.       Ganti nilai HISTSIZE dari 1000 menjadi 20

# $ HISTSIZE=20

# $ h

# Hasil:

# 

# 

# 

# 

# Analisa:

# HISTSIZE akan mengubah limit dari history terminal dari 1000 menjadi hanya 20 baris. Dengan perintah history, terlihat bahwa history terminal tidak melebihi 20 baris. Sebelum perintah h dieksekusi, perintah alias h=”history” dijalankan terlebih dahulu agar program perintah h tidak error.

# 

# b.      Gunakan fasilitas history dengan mengedit instruksi baris ke 5 dari instruksi yang terakhir dilakukan.

# $ !-5

# Hasil:

# 

# 

# Analisa:

# Perintah ini dimaksudkan untuk menjalankan perintah kelima dari akhir history. Karena perintah kelima dari akhir adalah “> maka perintah itu yang ditampilkan pada terminal. Perintah dengan history yang ada bisa dijalankan dengan !-[nomor history].

# 

# c.       Ulangi instruksi yang terakhir. Gunakan juga ^P dan ^N untuk bernavigasi pada history buffer

# $ !!

# Hasil:

# 

# 

# Analisa:

# Perintah !! berguna untuk menjalankan perintah terakhir yang dilakukan.

# 

# d.      Ulangi instruksi pada history buffer nomor tertentu, misalnya nomor 150

# $ !150

# Hasil:

# 

# 

# Analisa:

# Digunakan untuk menjalankan perintah sesuai nomor history.

# 

# e.       Ulangi instruksi dengan prefix “ls”

# $ !ls

# $ !?ls?

# Jelaskan perbedaan instruksi diatas

# Hasil:

# 

# 

# Analisa:

# Perintah ! ls digunakan untuk mengulangi instruksi dengan prefik. Perbedaan diantara 2 command tersebut adalah !ls akan menjalankan perintah ls yang terakhir kali dilakukan, sedangkan !?ls? akan mencari perintah terakhir yang mengandung kata kunci ls. Dalam case diatas hasilnya sama karena perintahnya bertepatan sama.

# 

# 5.      Prompt String (PS)

# a.       Edit file .bash\_profile, ganti prompt PS1 dengan ‘>’. Instruksi export diperlukan dengan parameter nama variab le tersebut, agar perubahan variable PS1 dikenal oleh semua shell PS1=’> ‘

# export PS1

# Eksperimen hasil PS1:

# $ PS1=“\! > “

# 69 > PS1=”\d > “

# Mon Sep 23 > PS1=”\t > “

# 10:10:20 > PS1=”Saya=\u > “

# Saya=stD02001 > PS1=”\w >”

# ~ > PS1=\h >”

# Hasil:

# 

# 

# 

# 

# 

# 

# Analisa: Untuk membuat dan mengedit file menggunakan editor, terdapat beberapa perintah yang dapat ditulis seperti nano, vi, dan vim. Namun, disini saya menggunakan perintah nano untuk membuat dan mengedit teks di editor. Setelah editor ditulis, untuk menyimpan filenya menggunakan Ctrl+O dan keluar menggunakan Ctrl+X. Bisa juga menyimpannya dengan cara Ctrl+X, tekan Y, lalu Enter. Setelah mengedit teks di editor, panggil perintah dengan menggunakan perintah PS1=”\!>”.

# 

# b.      Ubahlah warna shell prompt dengan warna biru dan berkedip.

# Hasil:

# 

# 

# Analisa:

# Membuat tulisan berwarna biru dan berkedip menggunakan 003 sebagai reset all, atribut 5 adalah agar nantinya berkedip, dan 34m adalah kode warna biru.

# 

# 6.      Bash script

# a.       Buat 3 buah script p1.sh, p2.sh, p3.sh dengan isi masing-masing:

# p1.sh

# #! /bin/bash

# echo “Program p1”

# ls –l

# p2.sh

# #! /bin/bash

# echo “Program p2”

# who

# p3.sh

# #! /bin/bash

# echo “Program p3”

# ps x

# 

# Hasil:

# 

# 

# 

# 

# 

# 

# 

# 

# Analisa:

# Untuk membuat dan mengedit file menggunakan editor, terdapat beberapa perintah yang dapat ditulis seperti nano, vi, dan vim. Namun, disini saya menggunakan perintah vi untuk membuat dan mengedit teks di editor. Untuk memulai ketikan, dengan cara menekan tombol i pada keyboard. Setelah editor ditulis, untuk menyimpan filenya kemudian keluar dengan menekan tombol esc, lalu ketik :wq. Keterangan menggunakan editor vi sebagai berikut:

# a = masuk ke menu insert / editing. ditandai dengan munculnya tulisan –INSERT– dibagian bawah layar.

# Esc (Escape) = keluar dari menu edit.

# :w = Save atau menyimpan hasil editing.

# :q = Keluar / exit.

# :wq = Save kemudian keluar.

# :q! = keluar tanpa save (abaikan perubahan file).

# /katayangdicari = mencari / search “katayangdicari”.

# :%s/katalama/katabaru = mengganti semua katalama dengan katabaru.

# 

# b.       Jalankan script tersebut sebagai berikut dan perhatikan hasilnya:

# $ ./p1.sh ; ./p3.sh ; ./p2.sh

# $ ./p1.sh &

# $ ./p1.sh $ ./p2.sh & ./p3.sh &

# $ ( ./p1.sh ; ./p3.sh ) &

# Hasil:

# $ ./p1.sh ; ./p3.sh ; ./p2.sh

# 

# 

# 

# 

# 

# 

# Analisa:

# Perintah di atas berfungsi untuk menjalankan program p1 kemudian dilanjutkan program p3 dan dilanjutkan program p2.

# 

# $ ./p1.sh &

# 

# 

# 

# 

# Analisa:

# Perintah di atas berguna untuk menjalankan isi file p1.sh yaitu ls –a, serta menjalankannya pada background.

# 

# $ ./p1.sh $ ./p2.sh & ./p3.sh &

# 

# 

# 

# 

# 

# 

# Analisa:

# Ketiga file tersebut akan dijalankan pada background sehingga memiliki pid masing” serta satu perintah yang telah seleasai dieksekusi yaitu file p1.sh yang sebelumnya dijalankan.

# 

# $ ( ./p1.sh ; ./p3.sh ) &

# 

# 

# 

# 

# 

# 

# Analisa:

# Dalam case ini p1 dan p3 akan dijalankan pada satu kesatuan perintah tetapi tetap menghasilkan output dari p1 dan p3. Atau dengan kata lain, perintah ini digunakan untuk menjalankan program p1 dan menjalankan p3 dan kemudian dibungkus di dalam background.

# 

# 7.      Jobs

# a.       Buat shell-script yang melakukan loop dengan nama pwaktu.sh, setiap 10 detik, kemudian menyimpan tanggal dan jam pada file hasil.

# #!/bin/bash

# while [ true ]

# do

# date >> hasil

# sleep 10

# done

# Hasil:

# 

# 

# 

# 

# Membuat file pwaktu.sh menjadi executable

# 

# 

# 

# 

# Analisa:

# Untuk membuat dan mengedit file menggunakan editor, terdapat beberapa perintah yang dapat ditulis seperti nano, vi, dan vim. Namun, disini saya menggunakan perintah vi untuk membuat dan mengedit teks di editor. Ketika file editor telah berhasil dibuat, maka perintah chmod +x pwaktu.sh digunakan untuk membua file skrip menjadi file executable. Perintah cat digunakan untuk melihat hasil pada pwaktu.sh, utput hasil dari perintah date akan di belokkan ke dalam file hasil, dan dapat dilihat bahwa setiap 10 detik, program berjalan dengan menyipan tanggal dan jam pada file hasil.

# 

# b.      Jalankan sebagai background; kemudian jalankan satu program (utilitas find) di background sebagai berikut:

# $ jobs

# $ find / -print > files 2>/dev/null &

# $ jobs

# Hasil:

# Kita menjalankan dengan perintah ./pwaktu.sh

# 

# 

# $ jobs

# 

# 

# $ find / -print > files 2>/dev/null &

# 

# 

# $ jobs

# 

# 

# Analisa:

# File pwaktu.sh akan dijalankan pada background, lalu ditambahkan perintah  find / -print > files 2>/dev/null & yang berguna untuk mencari seluruh file pada direktori / dan hasilnya dibelokkan ke dalam files dan errornya ke /dev/null. Sehingga akan ada 2 jobs yang dijalankan.

# 

# c.       Jadikan program ke 1 sebagai foreground, tekan ^Z dan kembalikan program tersebut ke background

# $ fg %1

# $ bg

# Hasil:

# 

# 

# Analisa:

# Perintah fg %1 maksudnya memindahkan program 1 yang ada di backgroundmenjadi berada di foreground. Dan untuk memindahkan kembali ke background, dihentikan dulu dengan Ctrl+Z. Perintah bg digunakan untuk mengembalikan proses 1 ke background.

# 

# d.      Stop program background dengan utilitas kill

# $ ps x

# $ kill [Nomor PID]

# Hasil:

# 

# 

# 

# 

# 

# 

# Analisa:

# Perintah ps x digunakan untuk melihat daftar proses dengan PID. Selanjutnya perintah kill digunakanuntuk menghentikan proses sesuai PID. Dan untuk memastikan proses telah dihentikan maka kita perintah jobs. Dan terlihat proses pwaktu.sh sudah terminated.

# 

# KESIMPULAN:

# Berdasarkan percobaan yang telah dilakukan, dapat diambil kesimpulan bahwa shell merupakan suatu program yang dibuat oleh user yang biasanya digunakan untuk memasukkan beberapa perintah secara bersamaan yang menunggu di compile dan akan dijalankan sesuai syntax yang ada dalam bash shell. Dalam penggunaannya, bash shell mempermudah user untuk melakukan pekerjaan, contohnya dengan adanya perintah history. History memudahkan user untuk mengedit kembali instruksi kompleks dan panjang, terutama bila terjadi kesalahan pada penulisan instruksi maupun parameter. Selain itu, ada pula bash script yang merupakan file yang berisi koleksi program yang dapat dieksekusi. Bash script juga dapat diedit menggunakan editor vi, nano, maupun vim. Dengan adanya editor, user bisa memanfaatkan proses pengeditan program pada satu layar penuh.